

AD-A242 210



DTIC

ELECTE

NOV 1 1991

Vign

**LEARN PROBABILITY DISTRIBUTIONS
WITH
THE CONTRASTIVE HEBBIAN ALGORITHM**
Technical Report AIP - 146

J. R. Movellan and J. L. McClelland

The Artificial Intelligence and Psychology Project

Departments of
Computer Science and Psychology
Carnegie Mellon University

Learning Research and Development Center
University of Pittsburgh

91-14672



01 10 31 041

**LEARN PROBABILITY DISTRIBUTIONS
WITH
THE CONTRASTIVE HEBBIAN ALGORITHM**
Technical Report AIP - 146

J. R. Movellan and J. L. McClelland

Department of Psychology
Carnegie Mellon University
Pittsburgh, PA 15213

July 9, 1991
version NC-1.3

This research was supported by the Computer Sciences Division, Office of Naval Research, under Contract Number N00014-86-K-0678. Reproduction in whole or in part is permitted for purposes of the United States Government. Approved for public release; distribution unlimited.

Accession For	
DTIC Unann	<input checked="" type="checkbox"/>
DTIC Tab	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution	
Availability Codes	
Dist	Avail and/or Special
A-1	

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified			1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; Distribution unlimited.	
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE				
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AIP-146			5. MONITORING ORGANIZATION REPORT NUMBER(S)	
6a. NAME OF PERFORMING ORGANIZATION Carnegie-Mellon University		6b. OFFICE SYMBOL (If applicable)		7a. NAME OF MONITORING ORGANIZATION Computer Sciences Division Office of Naval Research
6c. ADDRESS (City, State, and ZIP Code) Department of Psychology Pittsburgh, PA 15213			7b. ADDRESS (City, State, and ZIP Code) 800 N. Quincy Street Arlington, Virginia 22217-5000	
8a. NAME OF FUNDING / SPONSORING ORGANIZATION Same as Monitoring Organization		8b. OFFICE SYMBOL (If applicable)		9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-86-K-0678
8c. ADDRESS (City, State, and ZIP Code)			10. SOURCE OF FUNDING NUMBERS	
			PROGRAM ELEMENT NO. N/A	PROJECT NO. N/A
			TASK NO. N/A	WORK UNIT ACCESSION NO. N/A
11. TITLE (Include Security Classification) Learning probability distributions with the contrastive hebbian algorithm				
12. PERSONAL AUTHOR(S) Movellan, J. R. and McClelland, J. L.				
13a. TYPE OF REPORT		13b. TIME COVERED FROM _____ TO _____		14. DATE OF REPORT (Year, Month, Day) 1991, July 19
				15. PAGE COUNT 17
16. SUPPLEMENTARY NOTATION version NC-1.3				
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	SUB-GROUP		
19. ABSTRACT (Continue on reverse if necessary and identify by block number)				
<p>This paper presents a method for training connectional networks that adhere to the principles of graded, random, adaptive, and interactive propagation of information (GRAIN). While our analysis has been motivated by our desire to find a learning algorithm that would work in this environment, we have succeeded in implementing a model that encompasses a large class of previous connectionist algorithms under the same theoretical principles and that expands the scope of problems they can learn. Simulations show examples where GRAIN networks successfully approximate both discrete and continuous probability distributions, demonstrating that their scope extends beyond what can be learned by backpropagation networks or standard Boltzmann machines.</p>				
20. DISTRIBUTION / AVAILABILITY OF ABSTRACT <input type="checkbox"/> UNCLASSIFIED/UNLIMITED <input checked="" type="checkbox"/> SAME AS RPT <input type="checkbox"/> DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION	
22a. NAME OF RESPONSIBLE INDIVIDUAL Alan L. Meyrowitz			22b. TELEPHONE (Include Area Code) (202) 696-4302	22c. OFFICE SYMBOL N00014

Learning Probability Distributions with the Contrastive Hebbian Algorithm

J. R. Movellan and J. L. McClelland

Department of Psychology

Carnegie Mellon University

Pittsburgh Pa 15213

July 9, 1991 version NC-1.3

ABSTRACT

This paper presents a method for training connectionist networks that adhere to the principles of graded, random, adaptive, and interactive propagation of information (GRAIN). While our analysis has been motivated by our desire to find a learning algorithm that would work in this environment, we have succeeded in implementing a model that encompasses a large class of previous connectionist algorithms under the same theoretical principles and that expands the scope of problems they can learn. Simulations show examples where GRAIN networks successfully approximate both discrete and continuous probability distributions, demonstrating that their scope extends beyond what can be learned by backpropagation networks or standard Boltzmann machines.

1 INTRODUCTION

This paper presents a method for training connectionist networks that adhere to the principles of graded, random, adaptive and interactive propagation of information. These principles have emerged from issues in cognitive modeling and have been consolidated as a research program named GRAIN (McClelland, 1990) committed to modelling normal and disordered cognition. Our analysis has been motivated by our desire to develop a learning algorithm for the GRAIN environment that would allow learning probability distributions.

Consider a network presented with a sample S of exemplars taken from a joint probability distribution of input and teacher vectors

$$S = \{ (x_1, y_1), (x_2, y_2), \dots, (x_s, y_s) \} \quad (1)$$

where $x_i = (x_{i1} \dots x_{iI})$ is an input vector, and $y_i = (y_{i1} \dots y_{iO})$, the corresponding teacher vector. We will refer to the components of a teacher vector as *teacher units*.

One limitation of deterministic networks like standard backpropagation is that they can only generate a unique output vector for each input rather than

a distribution of output vectors. This unique output vector is usually an estimate of the low order statistics (e.g., expected values) of the distribution of teachers with information about the higher order statistics of this distribution being omitted. Furthermore, the error functions minimized by most learning algorithms are optimal when the teacher units are statistically independent but are not so appropriate when interdependencies appear. For instance, minimizing sum of squares, the error function typically used in backpropagation, converges to the conditional arithmetic means of each teacher unit and gives optimal results when the teachers are independent Gaussian variables. In the same fashion, minimizing cross-entropy, an error measure commonly used when the teachers are Boolean (0,1) variables, converges to the conditional probabilities of each teacher unit and is the best when each teacher unit is an independent Bernoulli random variable.

Omitting the higher order statistics of the distribution of teachers is beneficial when the problem at hand is well modelled by deterministic input-output functions with added independent noise components. On the other hand, this higher order information is necessary in other important situations. Consider for example a simple English to Spanish translation problem where the English word "olive" has two equally likely translations one of Latin root "oliva" and one of Arabic root "aceituna". If we were to use distributed representations for these words, backpropagation will learn the average representations of the two acceptable alternatives, a blend which is not acceptable. In cases like this we want to learn likelihoods of distributed patterns of activation rather than expected values of individual units.

If we want to learn arbitrary probability distributions, we need an error function capable of capturing high order statistics, and we also need randomness, one of the GRAIN principles. Boltzmann machines are intrinsically random, and their learning algorithm minimizes an error function called information gain (Ackley *et al.*, 1985) that describes the extent to which the distribution of patterns of activation produced by the network matches the distribution of patterns in the environment. Unfortunately, since randomness greatly reduces learning speed, this property of Boltzmann machines has often been neglected with more emphasis made to speed them up (e.g., using annealing schedules or mean field approximations).

What we present here are the results of our efforts to develop a learning algorithm that adheres to the GRAIN principles. We call the algorithm contrastive Hebbian learning (CHL), a name inspired in the work of Galland and Hinton (1989) with the deterministic Boltzmann machine. Contrastive Hebbian learning is a generalization of the Boltzmann learning algorithm (Ackley *et al.*, 1985) for the continuous stochastic case and it includes a large variety of previous connectionist learning algorithms under the same theoretical principles. In this paper we present a theoretical framework for understanding how the learning algorithm works, and a series of simulations where GRAIN networks success-

fully approximate discrete and continuous probability distributions of various types.

1.1 DETERMINISTIC SKELETON

Let $\mathbf{a} = [a_1, \dots, a_n]^T$ be a real value activation column vector. Let $\mathbf{W} = [\mathbf{w}_1, \dots, \mathbf{w}_n]$ be a real value matrix of connections, where each $\mathbf{w}_i = [w_{1,i}, \dots, w_{n,i}]^T$ is the fan-in column vector of connections to unit i . The default implementation of GRAIN uses the update equations of the interactive activation and competition model (McClelland and Rumelhart, 1981) :

$$\frac{da_i}{dt} = \lambda_i ((\max_i - a_i) \text{net}_i - d_i (a_i - \text{rest}_i)) ; \text{net}_i \geq \text{rest}_i \quad (2)$$

$$\frac{da_i}{dt} = \lambda_i ((a_i - \min_i) \text{net}_i - d_i (a_i - \text{rest}_i)) ; \text{net}_i < \text{rest}_i \quad (3)$$

where $\text{net}_i = \mathbf{a}^T \mathbf{w}_i$, \max_i is the maximum activation value for unit i , rest_i the activation when the net input is zero, \min_i the minimum activation value, d_i a positive constant named decay which controls the sharpness of the activation function, and $0 \leq \lambda_i \leq 1$ a constant which controls the speed of processing. Any other continuous Hopfield update equation (Hopfield, 1984) would also work ¹.

If the weight matrix is symmetric the equations implement a version of the continuous Hopfield model (Hopfield, 1984) with an associated Goodness function of the following form:

$$G = H - S \quad (4)$$

where

$$H = \frac{1}{2} \mathbf{a}^T \mathbf{W} \mathbf{a} \quad (5)$$

is the harmony or consistency between the network activations and the weight constraints. The stress

$$S = \sum_{i=1}^n d_i s_i \quad (6)$$

is a weighted sum of penalty terms, s_i , for the activations departing from rest value

$$s_i = (\max_i - \text{rest}_i) \log \left(\frac{\max_i - \text{rest}_i}{\max_i - a_i} \right) - (a_i - \text{rest}_i) ; a_i \geq \text{rest}_i \quad (7)$$

¹The present GRAIN implementation also has optional logistic and hyperbolic tangent update functions as defined in (Hopfield, 1984).

$$s_i = (\min_i - \text{rest}_i) \log \left(\frac{a_i - \min_i}{\text{rest}_i - \min_i} \right) + (a_i - \text{rest}_i); \quad a_i \leq \text{rest}_i \quad (8)$$

The decay parameters, d_i , weight the relative importance of stress vs. harmony for each unit, and correspond to the gain parameters if logistic equilibrium activation functions are used (Hopfield, 1984).

It is easy to show (Movellan, 1990) that these equations implement a version of the continuous Hopfield model with the activation vector asymptotically equilibrating at a local maximum of G .

1.2 THE RANDOM COMPONENT

Noise is injected to the deterministic skeleton of GRAIN by introducing a random vector ν to the net vector

$$\text{net} = \mathbf{a}^T \mathbf{W} + \nu \quad (9)$$

As a default the random vector is made of independent identically distributed zero mean Gaussian variables:

$$\nu \sim N(0, \sigma_{(t)}^2 \mathbf{I}) \quad (10)$$

where $\sigma_{(t)}^2$ stands for variance at time t . Formally, GRAIN is a Markovian diffusion process that optimizes the goodness function subject to the constraints imposed by the random vector.

1.2.1 Stochastic stability

The same way we study for the deterministic skeleton whether the activations stabilize, we may investigate in the stochastic case whether the probability distribution of activation states stabilizes over time and whether these stable points depend on the starting conditions. For simplicity we analyze this aspect by discretizing time and partitioning the activation states of each unit into m states over the min-max interval. Let $p_{ij}(t)$ be the probability of entering state j of the m^n possible states at the t^{th} transition given that the initial state is i . It is easy to show that GRAIN is a regular Markov process, and thus, by applying the Markovian basic limit theorem (Taylor & Karlin, 1984) it follows that:

(1) there exists a limiting distribution

$$\hat{p}_{ij} = \lim_{t \rightarrow \infty} p_{ij}(t) \quad (11)$$

(2) the distribution is unique and independent of the starting conditions

$$\hat{p}_j = \hat{p}_{ij}, \quad i = 1..m^n \quad (12)$$

(3) this limiting probability distribution equals the long run proportion of time that the process will be in each of the states.


This last aspect which is known as the ergodic property provides great flexibility for employing different strategies to estimate limiting distribution statistics of GRAIN networks.

2 TRAINING: THE CONTRASTIVE THEOREM

Training is based on the contrastive theorem, which was analyzed in a previous paper (Movellan,90) and that we state below.

2.1 Contrastive theorem

Theorem: Let $L(P)$ be a function of the m^n dimensional vector $P = \{P_1 \dots P_{m^n}\}$. Consider the following two cases where L is optimized as a function of P : 1) a (-) case where all the m^n P dimensions are free, and 2) a (+) case where some constraints have been imposed (e.g., some of the values of P are not allowed as solutions). Let $\hat{p}^{(-)}$ be a value of P that achieves $\hat{L}^{(-)}$, a maximum of L in the free case. Let $\hat{p}^{(+)}$ be a value of P that achieves $\hat{L}^{(+)}$, a maximum of L for the constrained case. Define the contrastive function $CF = \hat{L}^{(-)} - \hat{L}^{(+)}$. IF L has a unique maximum on the (-) case THEN it follows that: 1) $CF \geq 0$, and 2) $CF = 0$ if and only if $\hat{p}^{(-)} = \hat{p}^{(+)}$.

Proof: The set of P points accessible in the constrained (+) case is a subset of the set of points accessible in the free (-) case. Since $\hat{L}^{(-)}$, the maximum in the (-) case is unique any other L value must be smaller. That is, if $\hat{p}^{(+)} \neq \hat{p}^{(-)}$ then $\hat{L}^{(-)} > \hat{L}^{(+)}$ and $CF > 0$. The only case where $\hat{L}^{(-)} = \hat{L}^{(+)}$ is when $\hat{p}^{(+)} = \hat{p}^{(-)}$. 

We will use the contrastive theorem to develop the learning algorithm. P will represent the probability distribution of the m^n network states. In most cases we are interested in learning probability distributions conditional on input unit patterns. Using the same idea as in the Boltzmann machine, in the (-) case we will clamp the input units and let the rest of the network (hidden and output units) run free. In the (+) case we will further constrain the network by also clamping the output units to the environment. The learning algorithm is based on minimizing the contrastive function by modifying the weight and the decay parameters. If we get the contrastive function to be zero, then the probability distribution of the free and clamped cases must be the same and thus the network has learned.

2.2 Contrastive learning

Learning is defined as reproducing the probability distribution of the environment conditional on the input. We conjecture that the equilibrium distribution of GRAIN is maximizing expected goodness subject to a noise constraint. This can be expressed in a Lagrange form as

$$L(W, P) = \langle G(W, P) \rangle - \beta N(P) \quad (13)$$

where $P = \{P_1 \dots P_m\}$ is a probability distribution over the m^n network states, $\langle \rangle$ stands for expected value, $N(P)$ is a function expressing the noise constraint, and β is a Lagrange multiplier. The basic limit Markov theorem guarantees that the equilibrium distribution is unique. Thus, if our conjecture is true this distribution is the unique global maximum of L and we can apply the contrastive theorem on the L function: Define (+) as the case where inputs and outputs are clamped and (-) the case where only inputs are clamped. Define the contrastive function as

$$CF = \hat{L}^{(-)} - \hat{L}^{(+)} \quad (14)$$

Applying the *contrastive theorem* it follows that CF is always ≥ 0 and if $CF = 0$ the limiting distribution is the same in the (+) case, where inputs and outputs are clamped, as in the (-) case, where only inputs are clamped. Thus if $CF = 0$ we have successfully learned the desired probability distributions.

We can perform gradient descent on the contrastive function with respect to weights and with respect to decays. In order to do so we need the derivatives of the equilibrium point of L . Using the chain rule we may decompose these derivatives in two components²

$$\frac{d\hat{L}}{dw_{ij}} = \frac{\partial \hat{L}}{\partial w_{ij}} + \sum_{k=1}^{m^n} \frac{\partial \hat{L}}{\partial \hat{p}_k} \frac{\partial \hat{p}_k}{\partial w_{ij}} \quad (15)$$

It is easy to show that the first part of equation 15 is $\langle \hat{a}_i \hat{a}_j \rangle$. To first order the second part of equation 15 vanishes because we are at a stable point of L , where $\frac{\partial \hat{L}}{\partial \hat{p}_k} = 0$ for all k ³. Therefore, to first order

$$\frac{\partial CF}{\partial w_{ij}} = \langle \hat{a}_i^{(-)} \hat{a}_j^{(-)} \rangle - \langle \hat{a}_i^{(+)} \hat{a}_j^{(+)} \rangle \quad (16)$$

from which the rule for training weights is obtained. We will refer to this rule as *contrastive Hebbian learning*.

²A weight can influence the value of the L maximum by changing the equilibrium probability distribution of states and also by changing the goodness value of each state. The left side of the equation indicates the total derivative where both these influences are taken into consideration. The right side expresses these separate influences as two additive factors.

³For a similar version of this argument see (Hinton, 1989), where it is applied to the deterministic Boltzmann machine.

Similarly, to first order the derivatives for the decay terms are

$$\frac{\partial CF}{\partial d_i} = \langle \hat{s}_i^{(+)} \rangle - \langle \hat{s}_i^{(-)} \rangle \quad (17)$$

Moving in directions opposite to the derivatives we would obtain the appropriate learning rules.

2.3 Sampling strategies

Since the learning algorithm needs equilibrium distribution statistics, it is important to get these statistics in a fast and accurate way. One approach is to use annealing schedules by starting the settling process with a large noise component and gradually diminishing it. Another approach is to use sharpening or mean field annealing where initially large decay values are slowly replaced by smaller ones. Combinations of sharpening and annealing are also possible. Both sharpening and annealing schedules are oriented to sampling activation statistics when the network is visiting the attractor with biggest goodness value. Since the statistics of this attractor tend to dominate the desired equilibrium distribution, these procedures save time and in many cases provide accurate statistics. Unfortunately, these procedures may also run into problems when the network has to learn probability distributions where there is more than one equally desirable pattern of activation for the same input. In this case each of the desired patterns will have a corresponding maximum with the same goodness value. Since annealing schedules are designed to visit only one of the maxima at a time, the obtained statistics will be biased and will lead to instabilities in the learning process. In such cases, we have found it beneficial to let the network visit several large attractors by settling several times with different random starting values before changing the weights. Since the network is ergodic, equilibrium statistics using one or many restarts converge, but in practice we have found that they are obtained faster with the multiple restarts method. In our simulations we use the multiple restarts technique and do not use annealing or sharpening.

3 RELATION TO OTHER MODELS

GRAIN is currently implemented as a continuous Hopfield network with an added random component. As such, GRAIN is a Markovian diffusion process similar to that used by Ratcliff (1978) to model reaction time distributions in human cognition. GRAIN is also a variant of the Gaussian machine developed by Akiyama et al. to solve optimization problems, but contrary to Ratcliff's model or to the Gaussian machine, GRAIN can be trained. The name contrastive Hebbian learning was inspired by Galland and Hinton's reference to the contrastive Hebbian synapse in the deterministic Boltzmann machine (Galland and Hinton, 1989). The contrastive Hebbian learning algorithm for weights was

first applied by Hopfield to his discrete model but ~~not~~ theoretical ground was offered at the time (Hopfield *et al.*, 1983). A mathematical foundation based on statistical mechanics was given with the discrete stochastic Boltzmann machine (Ackley *et al.*, 1985). The algorithm was also derived from statistical mechanics principles for the continuous deterministic case (Peterson and Anderson, 1987; Peterson and Hartman, 1989; Hinton, 1989). Movellan (1990) noticed that CHL could be generalized, without reference to statistical mechanics, to any deterministic continuous Hopfield network by using the contrastive theorem. In this paper we generalize the learning algorithm for the continuous stochastic case and propose a rule for training the decay parameters. GRAIN encompasses a large set of models as special cases. By turning decays to zero, lambdas to 1, and noise to zero GRAIN becomes the schema model (Rumelhart *et al.* 1986). By adding decay terms, GRAIN implements the update equations of the interactive activation and competition (IAC) model (McClelland and Rumelhart, 1981). Both the schemata and the IAC model are versions of the continuous Hopfield model (Hopfield, 1984). If we add Gaussian noise we get a version of the Gaussian machine (Akiyama *et al.* 1989). If we use a logistic update function with large gain, lambdas turned to 1, and asynchronous activation update, the network approximates the discrete Hopfield model (Hopfield, 1982). If we then add Gaussian noise, the network approximates a Boltzmann machine (Ackley *et al.*, 1985). In all cases the networks can be trained with the same contrastive learning algorithm. ✓

4 SIMULATIONS

The purpose of the simulations is to investigate whether CHL can be used to train GRAIN in some standard problems as well as problems outside the scope of our current connectionist learning algorithms. In particular we show the results of the 4 following problems: 1) Standard XOR, 2) Translation problem, 3) Learning Independent Continuous probability distributions, 4) Learning continuous probability distributions governed by XOR. In all simulations the IAC update function was used with $\max = 1.0$, $\min = -1.0$, $\text{rest} = 0.0$. The weights were symmetric, and decays were maintained constant and equal for all units. Learning was done in epoch mode letting the network settle several times per pattern with different random starting values and accumulating the statistics for all the patterns before changing the weights. No annealing or sharpening schedules were used.

4.0.1 Standard XOR

The purpose of this simulation was to investigate whether GRAIN could be trained to solve a problem requiring hidden units. We performed 25 learning simulations with different random starting weights. The network consisted of

three layers (2 input units, four hidden units, 1 output unit) with connections only between adjacent layers. Within each layer the units were interconnected with self connections fixed to zero. Initial weights were sampled from a $(-1,1)$ uniform distribution. Learning was done in epoch mode with 10 settling restarts per pattern. Each settling consisted of 100 initial cycles of asynchronous activation update ⁴ where statistics were not collected, and 100 more cycles where statistics were collected. Decays were set at 0.1, lambdas at 0.1, and the Gaussian noise standard deviations were set at 1.0 for each unit. The stepsize constant for weight adjustment was set at 0.25. A 0.01 weight decay constant was also applied after each learning epoch. The median number of learning epochs until the information gain error measure was smaller than 0.04 was 26. This is a very strict learning criterion⁵. We then turned the network to $(-1, +1)$ Boltzmann mode by setting the activations to 1 if the net input combined with noise was greater than 1 or to -1 otherwise. In Boltzmann mode the median number of epochs was 65. The difference was statistically significant (Wilcoxon $p \leq 0.001$). We have tried a number of variations in the parameters and in all cases using graded activations was beneficial to speed up learning. The difference was less striking when less stringent learning criterion were used.

4.0.2 Translation problem

The purpose of the simulation was to investigate whether GRAIN can learn probabilistic mappings using distributed representations. The inspiration for this simulation was a problem that arises when training networks with distributed representations to do translations from one language to another. In those cases where a word has two or more acceptable translations, backpropagation converges to the expected values of each teacher unit, producing unacceptable blends of distributed representations. In our simulation "words" were encoded as random binary patterns distributed amongst 8 "English" and 8 "Spanish" units. There were 24 additional hidden units and all $24+8+8$ units were fully interconnected. Sometimes Spanish units were clamped to get a correct translation in the English module, and sometimes the English units were clamped to get a translation in the Spanish module (see Table 1). Initial weights were sampled from a $(-1,1)$ uniform distribution. Each settling consisted of 500 initial cycles of synchronous activation update ⁶ where statistics were not collected, and 200 more cycles where statistics were collected. Decays were set at 0.1, lambdas at 0.1, and the Gaussian noise standard deviations were set at 1.0 for each unit. The stepsize constant for weight adjustment was set at 0.01 for the first 200 epochs, 0.005 the next 200 epochs and 0.001 for the last 200 epochs. Learning was done in epoch mode with 10 settling restarts per pat-

⁴In asynchronous mode one randomly chosen unit is updated at a time. A cycle is defined as performing as many random updates as the number of units in the network.

⁵For an explanation of the learning criterion see the appendix.

⁶In synchronous mode all the units in the network are updated at the same time.

tern. The results with GRAIN after 600 learning epochs are in Table 1 which shows that a good approximation to the desired probabilities was obtained. Most importantly, for the ambiguous words, where more than one translation is possible, the network was near always in one of the correct alternatives and did not generate unacceptable blends. In principle this problem is also solvable in Boltzmann mode but we have not examined this case.

Input	Translation	
house	casa 1.000 [0.998]	
home	casa 1.000 [0.796]	
tomorrow	mañana 1.000 [0.898]	
later	mañana 1.000 [0.698]	
olive	aceituna 0.500 [0.399]	oliva 0.500 [0.595]
be	ser 0.500 [0.700]	estar 0.500 [0.298]
casa	house 0.500 [0.367]	home 0.500 [0.582]
mañana	tomorrow 0.500 [0.499]	later 0.500 [0.398]
aceituna	olive 1.000 [0.992]	
oliva	olive 1.000 [0.999]	
ser	be 1.000 [0.900]	
estar	be 1.000 [0.899]	

Table 1: Translation problem: Column one shows the input pattern and columns 2 and 3 the possible translations. The two numbers for each translation represent the desired probability and, in brackets, the obtained probability of the translations after 600 learning epochs. A pattern was considered correct if each output unit activation was within a 0.4 range of the desired value (-0.9 or $+0.9$).

4.0.3 Learning independent continuous probability distributions

This problem cannot be learned with backpropagation networks or with Boltzmann machines. The GRAIN network consisted of 5 output units connected to 10 fully interconnected hidden units. Self connections were allowed. Initial weights were sampled from a $(-1,1)$ uniform distribution. Each output unit was trained to reproduce a continuous probability distribution according to Table 2. Learning was done in epoch mode with 10 settling restarts per pattern. Each settling consisted of 20 initial cycles of asynchronous activation update where statistics were not collected, and 500 more cycles where statistics were collected. Decays were set at 0.1, lambdas at 0.1, and the Gaussian noise standard deviations were set at 1.5 for each unit. The stepsize constant for weight adjustment was set at 0.005.

Figure 1 shows 10,000 activation cycles of the 5 output units after 1000 learning epochs. Figure 2 shows the probability density functions of the first

Ouput unit	Distribution	Expected Value
1	Binomial	0
2	Constant	0
3	Uniform	0
4	Constant	-0.5
5	Binomial	-0.5

Table 2: Desired probability distributions for each of the 5 output units

three and the last two output units throughout learning. It can be seen that the output distributions successfully approximate the desired distributions given the constraints imposed by the injected noise. The pairwise correlations of the output unit activations after training were zero to the third decimal place.

4.1 Learning continuous probability distributions governed by XOR

This is a problem that cannot be learned with Boltzmann machines or back-propagation networks and that necessitates hidden units. There were 2 input units, 1 output unit, and 10 hidden units. Initial weights were sampled from a $(-1,1)$ uniform distribution. The network was fully interconnected, with self connections allowed. The probability distribution to be learned by the output unit depended on the input conditions as indicated in Table 3.

Input Units	Distribution	Expected Value
-1 -1	Constant	0
-1 1	Binomial	0
1 -1	Binomial	0
1 1	Constant	0

Table 3: Desired output probability distributions as a function of the input patterns.

Learning was done in epoch mode with 4 settling restarts per pattern. Each settling consisted of 20 initial cycles of asynchronous activation update where statistics were not collected, and 500 more cycles where statistics were collected. Decays were set at 0.1, lambdas at 0.1, and the Gaussian noise standard deviations were set at 1.0 for each unit. The stepsize constant for weight adjustment was set at 0.001. Figure 3 shows the results after 600 learning epochs. It can

be seen that the probability distribution of the activations conditional on the four input patterns successfully approximates the desired distributions given the constraints imposed by noise.

5 CONCLUSION

Backpropagation networks can only learn central tendencies of independent output units, not probability distributions over patterns of output units. Stochastic Boltzmann machines can in principle learn probability distributions but are limited to discrete binary values. GRAIN extends the scope of previous connectionist learning algorithms and encompasses a large category of them under the same theoretical framework. Our simulations show that GRAIN networks can be successfully trained to approximate probability distributions conditional on the input patterns.

We hope this work contributes to a renewal of interest in going beyond the learning of central tendencies to the modeling of distributions. Considerable work needs to be done. Analytically, it remains to be shown rather than just conjectured that GRAIN is maximizing a function of the form proposed in equation 13. Secondly, it would be helpful to link minimization of the contrastive function with minimization of other error functions. The contrastive theorem guarantees that when the contrastive function is zero we have learned to match the environmentally specified probability distributions exactly. Unfortunately the theorem does not say whether the relationship between the contrastive function and error functions such as information gain is monotonic. In practice we have found this to be the case, with both information gain and the contrastive function decreasing as learning progresses but we have not yet established this link analytically.

6 Appendix: Implementational details

- We have found it beneficial when using the update function of equations 2 and 3 to clip the activations to values smaller than max and bigger than min . This is done to avoid spurious oscillations that occur when the activations are close to extreme values when approximating a continuous time process with discrete time steps. In our simulations we did not let the activations grow bigger than $max - \lambda * decay$ or smaller than $min + \lambda * decay$.

- We have also found it beneficial to use non-extreme teacher values. For instance for the XOR and translation problems the teachers were set to either -0.9 or 0.9 instead of -1.0 or 1.0. For Boltzmann mode the teachers were (-1, +1).

- As discussed in Movellan 1990, gradient descent calls for the self connections to be changed at half the rate of the other weights. Our simulations followed this rule.

- The random restarts for each settling were done by reinitializing the activations to uniformly distributed random values within the allowable activation range.

- The error criterion to stop the training process is based on the following information gain function. This measure is defined as

$$\sum_{\text{patterns}} \int_{\text{desired states}} P_{\text{desired}} \log \frac{P_{\text{desired}}}{P_{\text{obtained}}} \quad (18)$$

Where P stands for probability density. This measure is the continuous version of the discrete information gain function used in discrete Boltzmann machines. In practice, we approximate the integral by defining a region surrounding each of the desired distributed states and assessing the actual probability that the activations fall within that region. For the XOR a unitwise tolerance of 0.4 was used. With targets set at ± 0.9 the activation of the output unit had to be above 0.5 for +0.9 teachers and below -0.5 for -0.9 teachers to fall within the target region. For the XOR simulation, a value of information gain of 0.04 corresponds quite closely to an average probability of 0.99 that the network is in the target state across the 10 restarts x 100 cycles x 4 patterns.

References

- [1] Ackley D, Hinton G and Sejnowski T (1985) A learning algorithm for Boltzmann machines. *Cognitive Science*, 9, 147-169.
- [2] Akiyama Y, Yamashita A, Kajiura M, Aiso H (1989) Combinatorial Optimization with Gaussian Machines, *Proceedings of the International Joint Conference on Neural Networks* 1, 533-540.

- [3] Hinton G E (1989) Deterministic Boltzmann Learning Performs Steepest Descent in Weight-Space, *Neural Computation*, 1, 143-150.
- [4] Hopfield J (1982) Neural Networks and Physical Systems with emergent collective computational abilities. *Proceedings of the National Academy of Science USA*, 79, 2254-2558.
- [5] Hopfield J, Feinstein D, Palmer R (1983) Unlearning has a stabilizing effect in collective memories. *Nature* 304, 158-159.
- [6] Hopfield J (1984) Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences U.S.A.*, 81, 3088-3092.
- [7] Galland C, Hinton G (1989) Deterministic Boltzmann Learning in Networks with Asymmetric Connectivity. University of Toronto. Department of Computer Science Technical Report. CRG-TR-89-6.
- [8] McClelland J, Rumelhart D (1981) An Interactive Activation Model of Context Effects in Letter Perception: Part 1. An account of Basic Findings. *Psychological Review*, 88, 5.
- [9] McClelland J (1990) Toward a theory of information processing in graded random interactive networks. To appear in *Attention and Performance*. Vol. 14.
- [10] Movellan J R (1990) Contrastive Hebbian learning in the continuous Hopfield model. in D Touretzky, J Elman, T Sejnowski, G Hinton: *Connectionist Models: Proceedings of the 1990 Summer School*. San Mateo, Morgan Kaufman.
- [11] Peterson C, Anderson J R (1987) A mean field theory learning algorithm for neural networks. *Complex Systems*, 1, 995-1019.
- [12] Peterson C, Hartman E (1989) Explorations of the Mean Field Theory Learning Algorithm. *Neural Networks*, 2, 475-494.
- [13] Ratcliff R (1978) A theory of memory retrieval *Psychological Review*, 95, 238-255.
- [14] Rumelhart D, Smolensky P, McClelland J L, and Hinton G: Schemata and sequential thought processes in PDP models. in in D. Rumelhart, & J. L. McClelland (1986) *Parallel distributed processing: Explorations in the microstructure of cognition. Volume 2: Psychological and biological models*. Cambridge, Mass: M. I. T. Press.
- [15] Taylor I, Karlim S (1984) An introduction to stochastic modeling. Orlando, Academic Press.

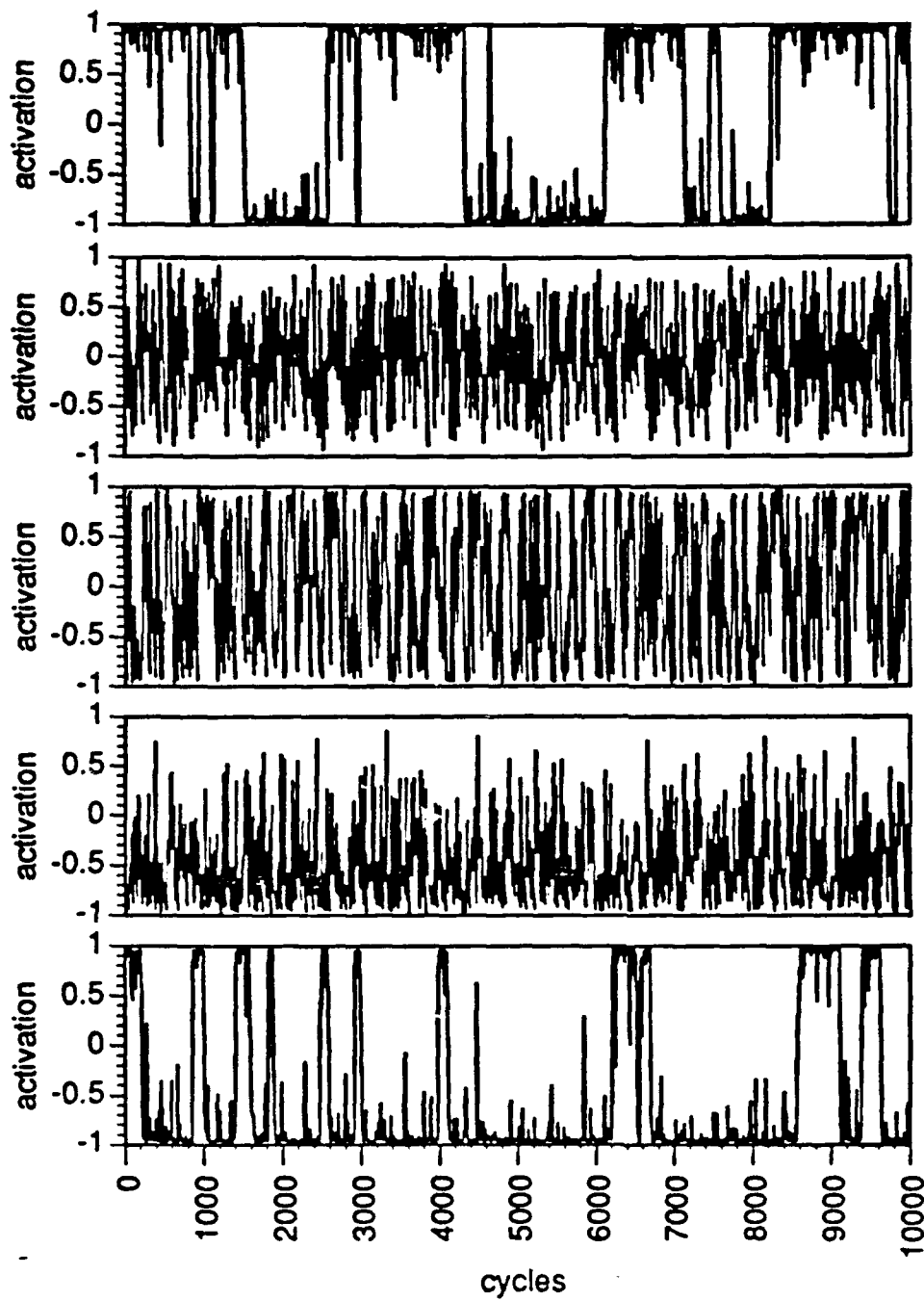


Figure 1: Output unit activations after 300 learning epochs. Each row represents the activation of one output unit throughout 10,000 cycles.

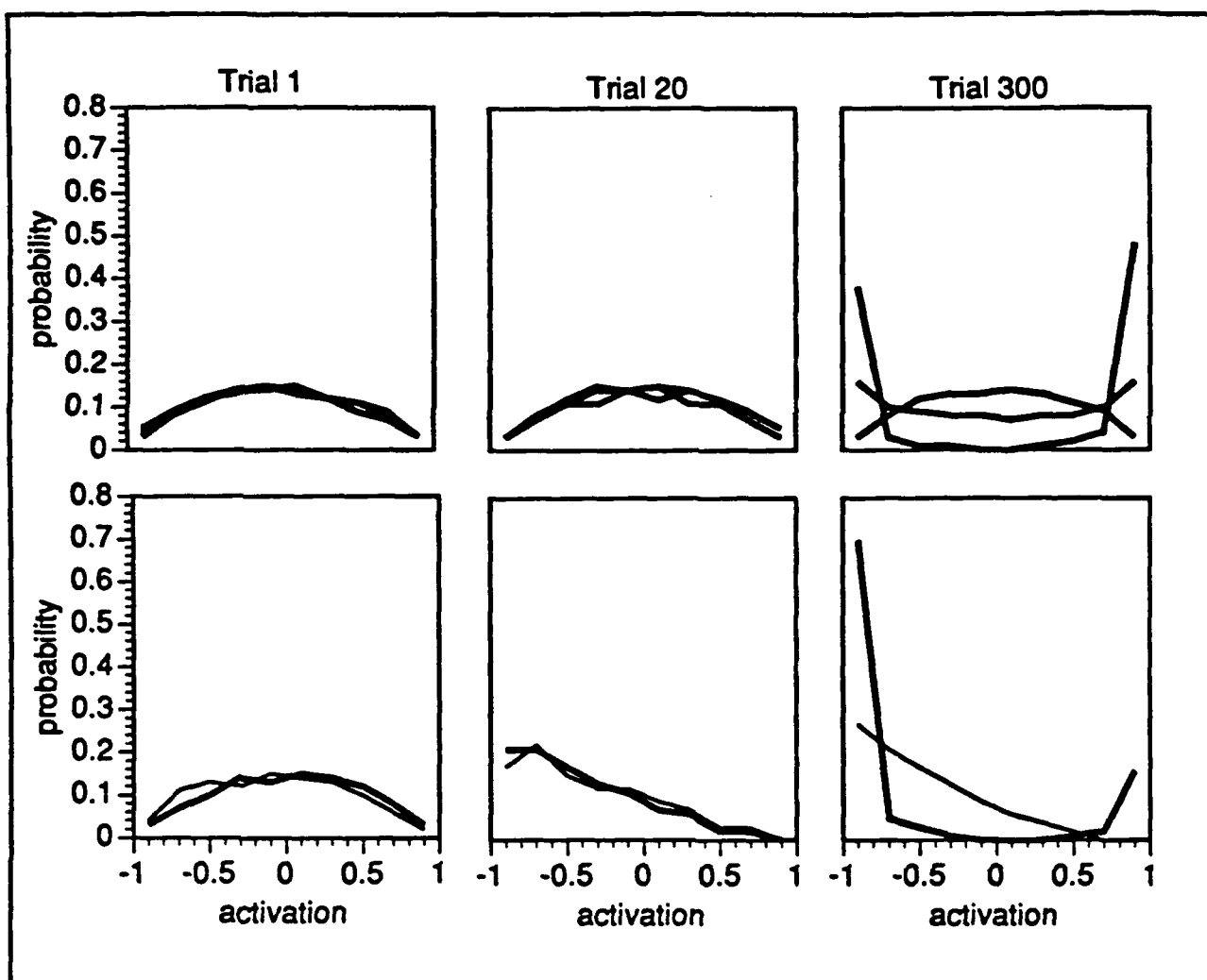


Figure 2: Probability distribution of the first three output units (top line); and the last two output units (bottom line) throughout learning.

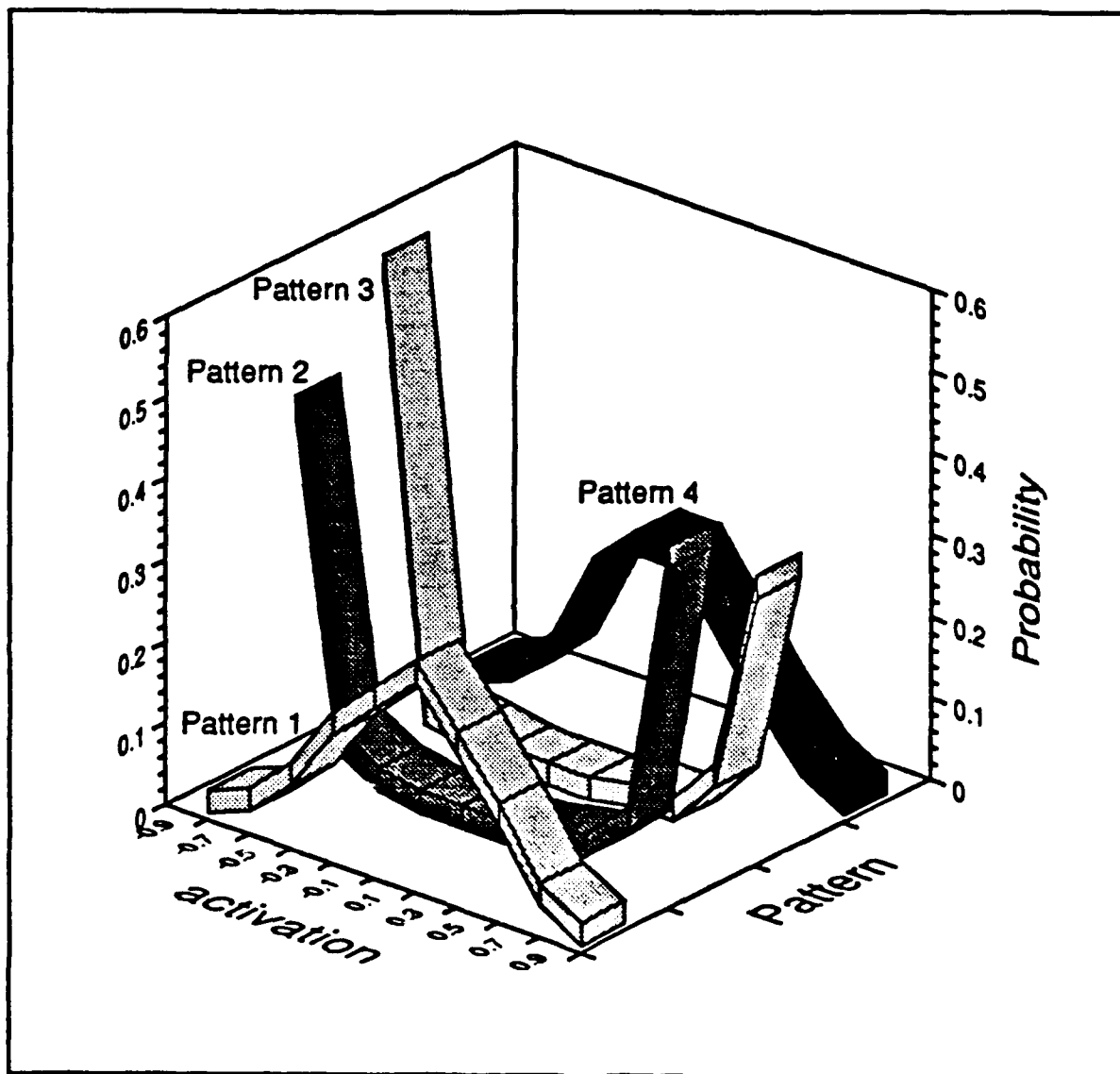


Figure 3: Probability distribution of the output unit conditional on the four input unit patterns.